March 2009

Friendly House Web Site

Paul Anthony Gibler
Worcester Polytechnic Institute

Friendly House IQP Report

Paul Gibler
Project #GFP0803

Advisor Signature: _____

Student Signature: _____

Table of Contents

Introduction

The Friendly House Interdisciplinary Qualifying Project (IQP) had a goal of creating an easy to maintain website for Friendly House, Inc., a food shelter located in Worcester, MA. The need for this arose since their web presence and site home page, as of March 9, 2009, amounted to an image of the front of the Friendly House building with their contact information written on it. This need was met by a team consisting of Dylan Streb and myself. I wrote the code for both displaying and maintaining the website, and Streb compiled a user guide for the administrators of the website at Friendly House.

The project itself went through several iterations. The first one did not initially include me, as I joined the project in the middle of its development process. Once I did join, my task was to use the current code base and website developed by the team and expand upon it so that it would be capable of handling all of the features the client would need. I did this over the first half of the summer of 2008, in which I added features such as dynamic loading of pages into the content section of the website, configurable page links, and made the news section simpler to maintain and update, by having all of the information loaded from plain text files and displayed on the web page after being processed by a PHP script. However, editing the individual pages still required the user to know HTML to update them, as they were loaded into the content section as-is. The initial solution that I decided upon was to create a site generation language to remove the need for developing the sites with HTML, as the users would then be able to use the site generation language and have the site manager application automatically generate the HTML from the language. This did not prove to be a significant enough upgrade over HTML as far as ease of use was concerned, and this part of the project was eventually scrapped.

In the next iteration of the project, I decided that a tool would be a good solution to solve this problem, and the web-based text editor FCKeditor was incorporated into the site management page to maintain individual web pages. An extension of jQuery, the jQuery File Tree, was used to navigate the individual pages of the website and select one for editing. While functional, the user experience was lacking, and serious renovations were necessary.

The only part of the project at this point that proved itself both easy to use and develop with was the FCKeditor tool. Since the site was going to be significantly renovated, I decided that the best way to handle this was to simply scrap everything unrelated to the FCKeditor and start fresh. This current iteration of the project was started at the beginning of Christmas break and has been ongoing since then. After some research, I decided to create a fully fledged Content Management System (CMS) for maintaining everything on the website, including but not limited to the web pages, page categories,

images, and footer. The CMS was written in a combination of PHP and JavaScript, with the PHP and JavaScript being used for the front-end and with the PHP also being used for the back-end, to interact with both the site pages and configuration files. A key goal of the CMS was to make it as easy to use as possible – this was accomplished by developing an intelligent design for the user interface in the CMS web application, by creating categories for related features so that navigating it was both simple and intuitive. Altering pages was also meticulously implemented. Users can edit page contents using a single feature, and edit the page title using another feature. By separating the features and only implementing the ones they would need, the hope was that users would not get lost trying to wade through features they didn't need or want. Based on the feedback from the clients during the demos, this appears to be a success.

The Initial Project

Overview

The initial project was a site displayer written in PHP that incorporated CSS for styling the pages of the web site. It included a pop-up menu bar that contained each program the clients at Friendly House wanted written for the web site. As well, it contained a section for displaying the contents of the web page and a news section that contained the current news of the day. Initially, the site displayer had several shortcomings such as having all of its content hard coded as well as being difficult to maintain from a development perspective that made it a decided failure in the end.

Capabilities

The site displayer had several useful capabilities, which found themselves in the final version of the project, albeit modified greatly. In the beginning, for a user to edit the content of individual pages, they would have to write the page using the Friendly House Site Generator language, a domain-specific language that was compiled into HTML. This feature is covered more in the next section of the paper. While it was not used in the end, from it came the most important feature and capability of the website, that is, allowing users to edit individual pages using FCKeditor, preview them, and save them on the website. This feature became a core component in the final iteration of the site manager tool.
As well, page content was loaded dynamically into the content section of the web page. The content loaded was determined by the GET variables of the HTTP request of the client program accessing the web page. The news section was loaded from a text file and rendered as is, meaning that the only way to incorporate formatting in the text of the web page was to wrap text in HTML tags. This was one of the things that the site maintainers at Friendly House explicitly stated that they did not want to have to deal with. This issue was handled by using FCKeditor to edit the content of web pages.

Shortcomings

The site displayer was written by the previous group in PHP, but the content of the site was hard coded into the page, making it hard coded and difficult to maintain. The user interface for the site manager was lacking, as it was unintuitive to use and was not pleasing to look at. The site displayer itself did not look good either, and the CSS written for the site displayer was poorly written, undocumented, and very difficult to maintain. I concluded that scrapping most of the parts of the initial project and starting anew would be the best direction for the IQP to go so that the clients would be satisfied.

Site Generation Language

Overview

The Site Generation Language (SGL) was a data language that contained Lisp-like syntax, and was compiled into an HTML page. It was designed with ease of use in mind, as the users of the site expressed their desire to avoid writing HTML. As an example, a user would type the following to generate bolded text in an HTML document:

```
<b>My bolded text</b>
```

In the SGL, a user would type the following to generate bolded text:

```
(bold My bolded text)
```

This simple example does not display the full capabilities of the language, but does show the inherent difference between the two languages, which is that the SGL is more verbose as far as how the text following the formatting code will be altered.

The code for the SGL compiler code can be found in Appendix Section I.

Two examples of SGL code and its HTML output can be found in the Appendix Section II.

Capabilities

The SGL contained several features that made it both easy to update, maintain, and use. Users could invoke the SGL compiler by pointing it to the path of the file to be compiled and an output HTML file, and it would generate the HTML file at the given location. It also had a schema file for all of the formatting codes, with both their inputs and outputs indicated at each formatting code.

The final schema file can be found in the Appendix Section III.

Shortcomings

The SGL suffered from several important shortcomings. Most importantly, it was not any easier or more readable than HTML for a user who lacked experience with HTML, and based on the level of development skill of the clients of the project, it was not reasonable to expect them to use it with any sort of success.

Friendly House Content Management System

Overview

The Friendly House Content Management System (FHCMS) is a tool developed for the clients at Friendly House to facilitate the ease and use of updating and maintaining pages, categories, images, and other artifacts that could appear on their web space. I wrote it in PHP, utilizing concepts first outlined by the previous iterations of the project, such as the use of configuration files for the settings of the website, and integrating FCKeditor into the FHCMS to make updating individual pages simple.

Open-Source Options

There are many open-source solutions for developers who would like to integrate a CMS into their web site. Two popular CMSs include Drupal and WordPress. The following is Drupal's developers' description of their software:

> Drupal is a free software package that allows an individual or a community of users to easily publish, manage and organize a wide variety of content on a website.

WordPress's developers describe their software as follows:

> WordPress was born out of a desire for an elegant, well-architectured personal publishing system built on PHP and MySQL and licensed under the GPL. It is the official successor of b2/cafelog. WordPress is fresh software, but its roots and development go back to 2001. It is a mature and stable product. We hope by focusing on user experience and web standards we can create a tool different from anything else out there.

Drupal, while powerful, contained a cumbersome interface that would be difficult to use for people not comfortable with other CMS software. This was the main reason Drupal was not selected as the CMS to use for the maintenance of Friendly House's web space. WordPress, on the other hand, has a very clear and concise interface. However, the software was geared towards maintaining and updating blogs, and was not intended to be used for web sites without some significant configuration and extension of the current code base. For these reasons, it was decided that developing a CMS tailor made for the users at Friendly House was the best direction for the project.

Capabilities

As it was tailor made for the clients at Friendly House, the FHCMS contained features that would facilitate them running their website more efficiently. It was designed so they would have to spend as little time navigating it, conversely letting them spend more time updating the site itself. An example of this includes the "Hiding" feature. Users can hide categories, which are just a collection of pages, and pages themselves. When a page or category is hidden, it does not appear in the navigation bar of the page. This is useful for seasonal web pages, such as the Thanksgiving Food Drive program that only appears during Thanksgiving. As well, the FHCMS contains many of the features standard to all CMS software, included but not limited to the ability to add and remove pages, edit the content of pages, add and remove categories, rename pages, rename categories, and update and change the footer.

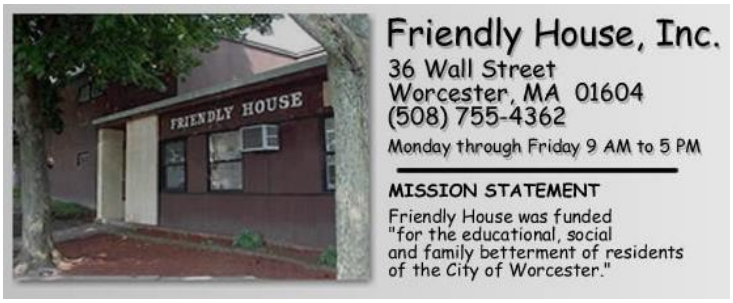
As far as systems capabilities goes, passwords are encrypted with 64-bit reverse string encoding to protect user login information. AJAX is used heavily to make the interface interaction with the back-end seamless, with PHP scripts being called to perform routines for the system.

Shortcomings

In its current state, the FHCMS can be considered a complete piece of software. However, there are several aspects of it that could be improved. The biggest shortcoming of the site is that all data is stored in text files in a hierarchal file structure. While this works, there are issues regarding the locking of files and corruption of data that would be much more easily handled with a true database such as MySQL or Oracle. As well, it lacks administrative tools for adding and removing users, as well as options such as disabling notifications. Despite these shortcomings, the clients at Friendly House should find the software to be both intuitive and easy to use, allowing them to update and maintain their site quickly and easily, which was the main goal of the IQP.

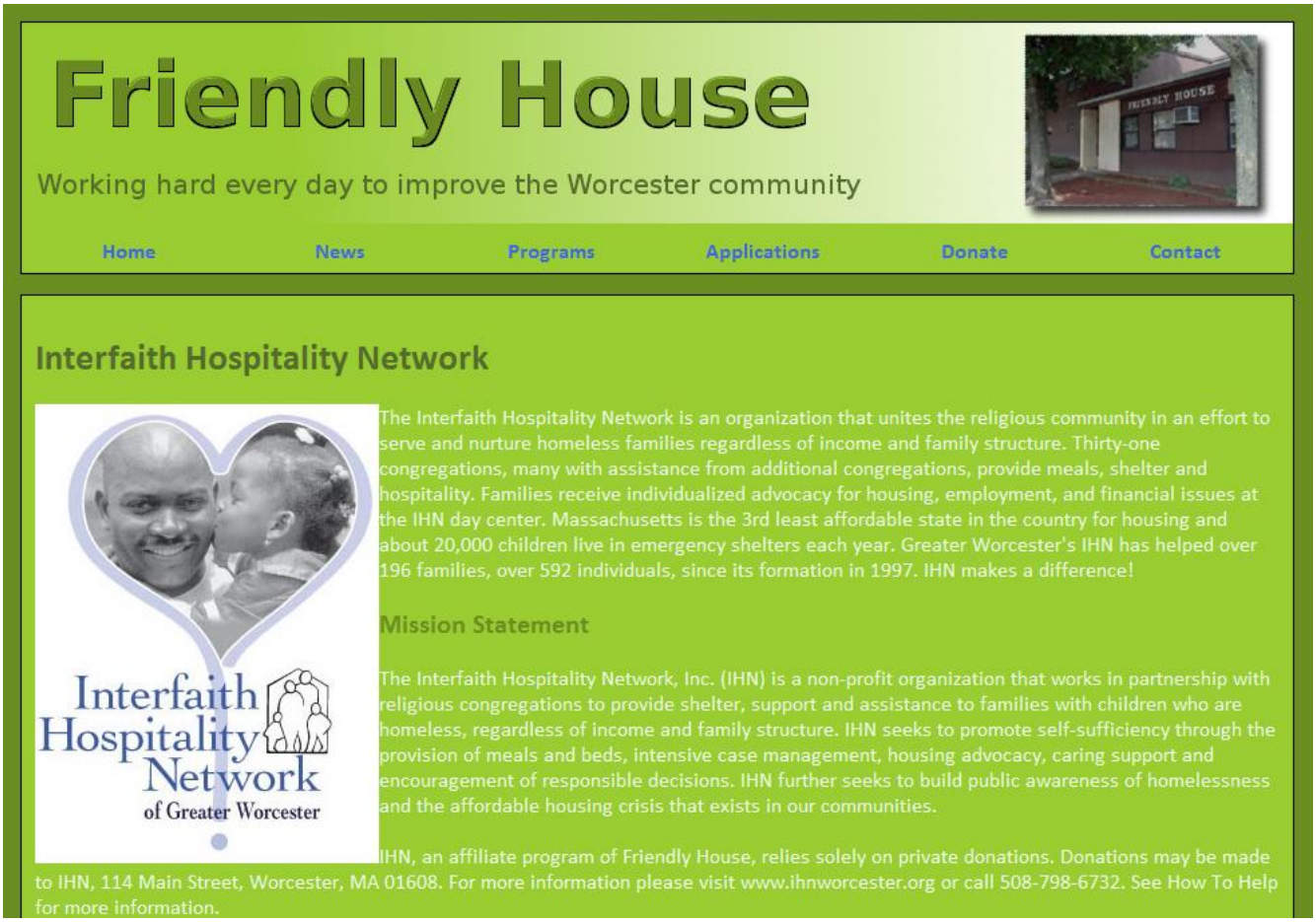
Before and After

The site has been improved dramatically from what they had previously. Before this IQP, their web presence amounted to a single image containing their contact information and an image of the Friendly House building at their domain name, http://friendlyhousema.org/.



Friendly House's Original Website

This image has been replaced with a proper website. One of the pages on it is as follows.



Like the original site, the contact information has been maintained on the website. It appears on both the front page and the footer. The footer is as follows:

Much like the rest of the website, this footer is alterable as well, satisfying the requirement of having the whole page be updateable.

Closing Words

The journey from the beginning of the IQP to its current state has been an interesting one. The project went through several iterations, some of which could have been avoided with better research from the project team. Using a CMS should have been the direction of the project since the beginning, but unfortunately much time was wasted upgrading the code from the previous project and then developing a data language. However, all of this work has taught me a lot. I got my first experience with developing a language, which was both interesting and eye opening. The CMS itself is a large-scale project which I am proud to say that I wrote and learned a lot from while developing it. I'm surprised by how well it turned out – it could have been a huge mistake writing a CMS, but I feel confident that it was the correct move based on our demos with the clients at Friendly House, as they have expressed that it looks easy enough for them to use. If they truly are happy with the final iteration of the project, then it's fair to say that we satisfied the customers' expectations, which is the true barometer of this project's success.

Appendix

Section I

```
#!/usr/bin/perl

use strict;
use Data::Dumper;
use Getopt::Long;
use FindBin '$Bin';

use lib "$Bin/lib";
use Parse::RecDescent;
use Config::Scoped;
use UNIVERSAL::isa;

my %in_cfg;
my %cfg = ();
my @tokens = ();

my $input_file;
my $output_file;
my $config_file = "schema.cfg";

Getopt::Long::Configure ("bundling", "auto_help");
if (!GetOptions(
                        "config_file=s"             => \$config_file,
                        "help"                  => sub {&usage},
                    ))
{
      &usage;
}

%in_cfg = %{ Config::Scoped->new(
                                                    file => $config_file,
                                                    warnings => { permissions =>
'off' },
                                                )->parse() };

$input_file = shift || "source/home.txt";
my $input_file_name = ($input_file =~ /\/(.*)$/)[0];
$output_file = shift || "output/$input_file_name.html";

my $string = '';
open INPUT_FILE, $input_file;
for(<INPUT_FILE>)
{
      $string .= "$_";
}
close INPUT_FILE;

$Parse::RecDescent::skip = '';

# Create and compile the source file
my $parser = Parse::RecDescent->new(q(
      startrule : field(s)
                        {     main::capture( $item[1] ); }
```

```
	ws :		/\s+/
	pws :		/\s*/
	field :			generator
				| paren_text
				| text
				| ws
	string_field:generator
				| paren_text
				| string_text
				| ws
	paren_text: '\(' field(s) '\)'
				{ $return = $item[1]."@{$item[2]}".$item[3] }
	string :	/\'/ string_field(s) /\'/
				{ $return = "@{$item[2]}"; }
	string_text :/[^\(\)\s\']+/
	text :			/[^\(\)\s]+/
	generator:	'\(' pws command pws '\)'
				{ $return = main::generate( $item{command} ); }
	command:	operation(s /\s+/) /\s*/ parameter(s /\s*/)
				{ $return = { instructions => $item[1], parameters =>
$item[3] }; }
				| operation(s /\s+/)
				{ $return = { instructions => $item[1], parameters => [()]
};	}
	parameter : generator
				| string
				| text
	operation : '-' text
				{ $return =  $item[2]; }
));

if( $parser->startrule($string) )
{
#	print "Valid.\n\n";

	open OUTPUT_FILE, ">$output_file";

	print OUTPUT_FILE combine_all( @tokens );
}
else
{
	print "Invalid file formatting\n";
}

sub capture
{
	@tokens = @{ (shift) };
}

sub combine_all {
	my @toks = @_;
	my $output = "";
	for my $tok (@toks)
	{
		$tok =~ s/\n/\n<br>/g;
		$output .= "$tok";
	}
	return $output;
}
```

```
sub generate
{
	my %command = %{ (shift) };
	# The command instructions.
	my @instructions = @{$command{instructions}};
	# The command parameters.
	my @p = @{$command{parameters}};
	my $parameter_count = @p;

	my %tag_group = ();

	# Load up all instructions with the number of parameters equal to the
parameter count.
	if(defined $in_cfg{parameter_count}{$parameter_count})
	{
		%tag_group = %{$in_cfg{parameter_count}{$parameter_count}};
	}
	# Then load up all instructions that take 'n' parameters.
	for my $instruction (keys %{$in_cfg{parameter_count}{n}})
	{
		$tag_group{$instruction}{body} =
$in_cfg{parameter_count}{n}{$instruction}{body};
		if(defined $in_cfg{parameter_count}{n}{$instruction}{pre})
		{
			$tag_group{$instruction}{pre} =
$in_cfg{parameter_count}{n}{$instruction}{pre};
		}
		if(defined $in_cfg{parameter_count}{n}{$instruction}{post})
		{
			$tag_group{$instruction}{post} =
$in_cfg{parameter_count}{n}{$instruction}{post};
		}
	}

	# Preprocess the text.
	for my $instruction (keys %tag_group)
	{
		if(defined $tag_group{$instruction}{pre})
		{
			eval($tag_group{$instruction}{pre});
		}
	}

	my $output = "";
	my $result = "";
	my $curr_instruction = "";
	# Output the body in the order that the instructions work.
	for my $instruction (@instructions)
	{
		if(defined $tag_group{$instruction}{body})
		{
			$curr_instruction = $tag_group{$instruction}{body};

			# Parameters with a count of 1 can be linked together.
			if($parameter_count == 1)
			{
				if(!$result)
				{
```

```
                        $result = $curr_instruction;
                    }
                    else
                    {
                        $result =~ s/\$p\[0\]/$curr_instruction/g;
                    }
                }
                else
                {
                    $result .= $curr_instruction;
                }
            }
            else
            {
                print "Instruction $instruction does not take $parameter_count
parameters:\n";
                for my $p (@p)
                {
                    print "$p\n";
                }
                print "Discontinuing parsing.\n";
                exit(0);
            }
        }

        $output .= $result;

        # Process the parameters.
        my $i = 0;
        for my $param (@p)
        {
            if( is_generator( $param ) )
            {
                $param = generate( $param );
            }
            $output =~ s/\$p\[$i\]/$param/g;
            $i++;
        }

        $output =~ s/\@p/@p/g;

        # Postprocesses the text.
        for my $instruction (keys %tag_group)
        {
            if(defined $tag_group{$instruction}{post})
            {
                eval($tag_group{$instruction}{post});
            }
        }

        return $output;
}

sub is_generator
{
        my $ref = shift;
        my $ishash = UNIVERSAL::isa($ref, "HASH");
        return $ishash;
}
```

Section II

Input)

```
(-article_title 'Friendly House Shelter')

(-section Description: )

The Friendly House Family Shelter is a short-term emergency shelter for homeless
families with children, with space for up to 13 families.

(-section 'How to apply:')

Applicants must be referred by DTA and have an interview with the shelter
supervisor. For more information, call or visit:

Address: 87 Elm Street
Worcester, MA 01609                (-link
'http://maps.yahoo.com/py/maps.py?&addr=87+Elm+St.&csz=Worcester%2C+MA+01609' 'Map
of this location')
Telephone: 508-792-1799
Office hours: 24 hours, every day of the year

Languages: English, Spanish
Transportation: Near a bus line

(-section Eligibility:)

This shelter is for women and children only. Families must be homeless and have a
referral from the DTA
```

Output)

```
<font size="6" color="#333366"><b>Friendly   House   Shelter</b></font><hr
width=300 \>
<br>
<br><font size="5" color="#333399"><b>Description:</b></font>
<br>
<br>The Friendly House Family Shelter is a short-term emergency shelter for
homeless families with children, with space for up to 13 families.
<br>
<br><font size="5" color="#333399"><b>How   to   apply:</b></font>
<br>
<br>Applicants must be referred by DTA and have an interview with the shelter
supervisor. For more information, call or visit:
<br>
<br>Address: 87 Elm Street
<br>Worcester, MA 01609                <a
href=http://maps.yahoo.com/py/maps.py?&addr=87+Elm+St.&csz=Worcester%2C+MA+01609>Ma
p   of   this   location</a>
<br>Telephone: 508-792-1799
<br>Office hours: 24 hours, every day of the year
<br>
<br>Languages: English, Spanish
<br>Transportation: Near a bus line
<br>
<br><font size="5" color="#333399"><b>Eligibility:</b></font>
<br>
```

```
<br>This shelter is for women and children only. Families must be homeless and have
a referral from the DTA
```

Input containing lists)

```
(-article_title 'Quinsigamond Village Community Center' )
The Quinsigamond Village Community Center is a multi-service neighborhood center.
The center is located at:

( -unordered_list
    ( -list_item   'Quinsigamond Village Community Center
                   16 Greenwood Street
                   Worcester, MA 01607 (-link
'http://maps.yahoo.com/py/maps.py?&addr=16+Greenwood+St.&csz=Worcester%2C+MA+01607'
'Map of this location')
                   Telephone: 508-755-7481

                   Hours: 9 AM to 4 PM, Monday through Friday')
)
(-line_break)
(-section 'Programs and services:')

The Quinsigamond Village Community Center offers the following programs and
services:

( -unordered_list
    (-list_item 'Senior Wellness Program')
    (-list_item 'Food pantry
                 Food distribution is Monday through Friday from 10:00 AM to 4:00
PM.')
    (-list_item 'Holiday assistance
                 Food baskets and Christmas toys are given out.')
    (-list_item 'Instructional baseball/basketball league
                 For boys and girls ages 5 through 12, November through March')
    (-list_item 'Worcester Intertribal Indian Center (WIIC)')
    (-list_item 'Employment services
                 Interviewing skills, application assistance, employment follow-up,
job crisis intervention')
    (-list_item 'Housing assistance')
    (-list_item 'After-school center
                 This is a 30-week long program made available through a grant from
Project Bread:
                 (-unordered_list
                     (-list_item 'ages 5 and 6 (limited to 20 children)' )
                     (-list_item 'Tuesday through Friday, 2:30 PM to 4:30 PM' )
                     (-list_item 'snacks are served')
                     (-list_item 'homework club')
                     (-list_item 'fun time')
                 )'
   )
    (-list_item 'Girl Scouts
                 Mondays, 2:30 PM to 4:00 PM')
    (-list_item 'Self-help groups:
                 Narcotics Anonymous (Fridays, 6 PM to 7 PM)
                 Alcoholics Anonymous (Saturdays, 2:30 PM to 5:00 PM)')
    (-list_item 'Neighborhood Watch
                 Meets on the 4th Tuesday of the month, from 4 PM to 5:30 PM')
    (-list_item 'Community meeting space
                 Call for availability')
```

)

Output containing lists)

```
<font size="6" color="#333366"><b>Quinsigamond Village Community
Center</b></font><hr width=300 \>
<br>The Quinsigamond Village Community Center is a multi-service neighborhood
center. The center is located at:
<br>
<br><ul><li>Quinsigamond Village Community Center
<br>                    16 Greenwood Street
<br>                    Worcester, MA 01607 <a
href=http://maps.yahoo.com/py/maps.py?&addr=16+Greenwood+St.&csz=Worcester%2C+MA+01
607>Map of this location</a>
<br>                    Telephone: 508-755-7481
<br>
<br>                    Hours: 9 AM to 4 PM, Monday through
Friday</li></ul>
<br><hr width=300 />
<br><font size="5" color="#333399"><b>Programs and services:</b></font>
<br>
<br>The Quinsigamond Village Community Center offers the following programs and
services:
<br>
<br><ul><li>Senior Wellness Program</li> <li>Food pantry
<br>                    Food distribution is Monday through Friday from
10:00 AM to 4:00 PM.</li> <li>Holiday assistance
<br>                    Food baskets and Christmas toys are given
out.</li> <li>Instructional baseball/basketball league
<br>                    For boys and girls ages 5 through 12,
November through March</li> <li>Worcester Intertribal Indian Center
(WIIC)</li> <li>Employment services
<br>                    Interviewing skills, application assistance,
employment follow-up, job crisis intervention</li> <li>Housing
assistance</li> <li>After-school center
<br>                    This is a 30-week long program made available
through a grant from Project Bread:
<br>                    <ul><li>ages 5 and 6 (limited to 20
children)</li> <li>Tuesday through Friday, 2:30 PM to 4:30 PM</li>
<li>snacks are served</li> <li>homework club</li> <li>fun
time</li></ul></li> <li>Girl Scouts
<br>                    Mondays, 2:30 PM to 4:00 PM</li> <li>Self-help
groups:
<br>                    Narcotics Anonymous (Fridays, 6 PM to 7 PM)
<br>                    Alcoholics Anonymous (Saturdays, 2:30 PM to 5:00
PM)</li> <li>Neighborhood Watch
<br>                    Meets on the 4th Tuesday of the month, from
4 PM to 5:30 PM</li> <li>Community meeting space
<br>                    Call for availability</li></ul>
```

Section III

```
{
	parameter_count 0
	{
		line_break =>
		{
			body = '<hr width=300 />';
		}
	}

	parameter_count 1
	{
		bold =>
		{
			body = '<b>$p[0]</b>';
		}
		center =>
		{
			body = '<center>$p[0]</center>';
		}
		italicize =>
		{
			body = '<i>$p[0]</i>';
		}
		link =>
		{
			body = '<a href=$p[0]>$p[0]</a>';
		}

		image =>
		{
			body = '<img src=$p[0]>';
		}

		list_item =>
		{
			body = '<li>$p[0]</li>';
		}
```

```
            article_title =>
            {
                  body = '<font size="6" color="#333366"><b>$p[0]</b></font><hr
width=300 \>';
            }

            section =>
            {
                  body = '<font size="5" color="#333399"><b>$p[0]</b></font>';
            }

            email =>
            {
                  body = '<a href="mailto:$p[0]">$p[0]</a>';
            }

            print =>
            {
                  body = '$p[0]';
            }
      }
      parameter_count 2
      {
            link =>
            {
                  body = '<a href=$p[0]>$p[1]</a>';
            }

            image =>
            {
                  body = '<img src=$p[1] align=$p[0]>';
                  pre =  'if( $p[0] =~ /[-_\s]?align[-_\s]?/ ) { $p[0] =~ s/[-
_]?align[-_]?//; };'
            }

            list_item =>
            {
                  body = '<li>$p[0]</li>$p[1]';
            }
```

```
	}

	parameter_count n
	{
		unordered_list =>
		{
			body = '<ul>@p</ul>';
		}

		list =>
		{
			body = '<ol>@p</ol>';
		}

		ordered_list =>
		{
			body = '<ol>@p</ol>';
		}

		definitions_list =>
		{
			head = '<dl>@p</dl>';
		}

		list_item =>
		{
			body = '<li>@p</li>';
		}
	}
}
```